

# PATENT ABSTRACTS OF JAPAN

(11)Publication number : 2000-332995

(43)Date of publication of application : 30.11.2000

(51)Int.Cl.

H04N 1/387

G06T 5/00

H04N 1/407

(21)Application number : 11-146593

(71)Applicant : RICOH CO LTD

(22)Date of filing : 26.05.1999

(72)Inventor : SAKUYAMA HIROYUKI

(30)Priority

Priority number : 11073321

Priority date : 18.03.1999

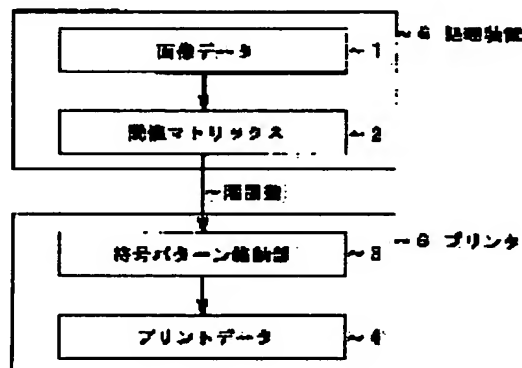
Priority country : JP

## (54) METHOD FOR GENERATING IMAGE DATA AND RECORDING MEDIUM

(57)Abstract:

**PROBLEM TO BE SOLVED:** To simultaneously perform halftone processing and resolution conversion in which raster development and the resolution conversion of an image are high speed and halftone processing, transfer data quantity and image quality are intermediate.

**SOLUTION:** Plural thresholds are sorted in order of values and arranged in each pixel of a threshold matrix 2. Each pixel of image data 1 is compared with the plural thresholds in the elements of a matrix corresponding to the pixel, and a pixel value is converted into the number of gradations. A code pattern 3 corresponding to the pixel and the number of gradations is selected, and print data 4 with high resolution is outputted according to them.



## LEGAL STATUS

[Date of request for examination]

27.03.2003

[Date of sending the examiner's decision of rejection]

[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]

[Date of final disposal for application]

[Patent number]

3688938

[Date of registration]

17.06.2005

[Number of appeal against examiner's decision of rejection]

[Date of requesting appeal against examiner's  
decision of rejection]

[Date of extinction of right]

(19) 日本国特許庁 ( J P )

(12) 公 開 特 許 公 報 ( A )

(11) 特許出願公開番号

特開2000-332995

( P2000-332995A )

(43) 公開日 平成12年11月30日 (2000. 11. 30)

(51) Int.Cl. <sup>7</sup>	識別記号	F I	テ-マコ-ト*(参考)	
H 0 4 N 1/387	1 0 1	H 0 4 N 1/387	1 0 1	5 B 0 5 7
G 0 6 T 5/00		G 0 6 F 15/68	3 1 0 J	5 C 0 7 6
H 0 4 N 1/407		H 0 4 N 1/40	1 0 1 E	5 C 0 7 7

審査請求 未請求 請求項の数13 O L (全 23 頁)

(21) 出願番号 特願平11-146593

(22) 出願日 平成11年 5 月26日 (1999. 5. 26)

(31) 優先権主張番号 特願平11-73321

(32) 優先日 平成11年 3 月18日 (1999. 3. 18)

(33) 優先権主張国 日本 ( J P )

(71) 出願人 000008747

株式会社リコー

東京都大田区中馬込 1 丁目 3 番 6 号

(72) 発明者 作山 宏幸

東京都大田区中馬込 1 丁目 3 番 6 号 株式  
会社リコー内

(74) 代理人 100073760

弁理士 鈴木 誠 (外 1 名)

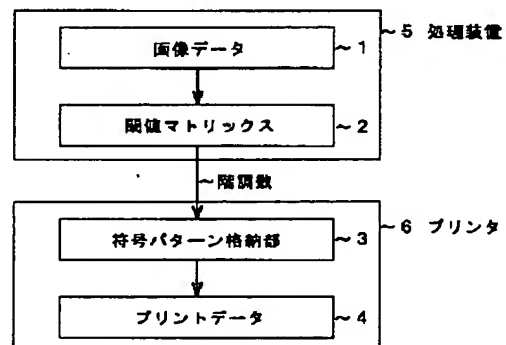
最終頁に続く

(54) 【発明の名称】 画像データ生成方法および記録媒体

(57) 【要約】

【課題】 ラスター展開とイメージの解像度変換が高速で、中間調処理、転送データ量、画質が中程度である、中間調処理と解像度変換を同時に行う。

【解決手段】 閾値マトリックス 2 の各要素には、複数の閾値が値順にソートされて配置されている。画像データ 1 の各画素と、画素に対応したマトリックスの要素内の複数の閾値とを比較して、画素値を階調数に変換する。画素と階調数に対応した符号パターン 3 を選択し、これにより、解像度の高いプリントデータ 4 を出力する。



## 【特許請求の範囲】

【請求項 1】 第 1 の解像度の画像データから第 1 の解像度より高い第 2 の解像度の画像データを生成する画像データ生成方法であって、前記第 1 の画像データの注目画素毎に、該画素の位置に対応した、ソートされた複数の閾値を用いて、画素値を階調数に変換し、前記画素の位置および前記変換された階調数に対応した所定の符号パターンを参照することにより、前記第 2 の画像データを生成することを特徴とする画像データ生成方法。

【請求項 2】 前記階調数変換時に誤差拡散処理を適用することを特徴とする請求項 1 記載の画像データ生成方法。

【請求項 3】 同一階調値における前記符号パターンを、前記第 1 の画像データの画素位置に対応して配置したとき、該配置がドット分散型となることを特徴とする請求項 1 記載の画像データ生成方法。

【請求項 4】 同一階調値における前記符号パターンを、前記第 1 の画像データの画素位置に対応して配置したとき、該配置がブルーノイズ型となることを特徴とする請求項 1 記載の画像データ生成方法。

【請求項 5】 前記階調数変換をホスト側で行い、前記階調数を画像出力装置に対して送信し、前記階調数の送信が  $n$  進法で行われるとき、前記階調数を  $n$  のべき乗とすることを特徴とする請求項 1 記載の画像データ生成方法。

【請求項 6】 第 1 の解像度の画像データから第 1 の解像度より高い第 2 の解像度の画像データを生成する画像データ生成方法において、前記第 1 の画像データの注目画素毎に、該画素の位置に対応した、ソートされた複数の閾値を用いて、画素値を階調数に変換し、前記画素の位置および前記変換された階調数に対応した所定の第 1 の符号パターンを参照することにより、前記第 2 の画像データを生成する画像データ生成方法であって、前記注目画素に対応した符号パターンおよび所定個数の前記第 1 の画像データに対応した符号パターンから構成される所定形状の注目符号パターン群と、特定符号パターン群とを比較し一致したとき、前記注目符号パターン群の所定の第 2 の符号パターンを、第 3 の符号パターンに変更することを特徴とする画像データ生成方法。

【請求項 7】 前記階調数変換をホスト側で行い、前記階調数を画像出力装置に対して送信し、該画像出力装置は前記階調数に対応した符号パターンを参照することにより前記第 2 の画像データを生成すると共に、前記第 3 の符号パターンへの変更を行うことを特徴とする請求項 6 記載の画像データ生成方法。

【請求項 8】 前記変更の対象となる前記第 3 の符号パターンは、前記所定の第 1 の符号パターンの何れとも異なることを特徴とする請求項 6 記載の画像データ生成方法。

【請求項 9】 前記注目符号パターン群を複数用いて、

該複数の注目符号パターン群と特定符号パターン群とを比較することを特徴とする請求項 6 記載の画像データ生成方法。

【請求項 10】 前記第 1 の画像データが複数の色成分を有し、該色成分毎に、前記所定形状の注目符号パターン群と特定符号パターン群とを比較するとき、前記色成分の内、少なくとも 1 つの色成分については、前記特定符号パターン群の数を、他の色成分の特定符号パターン群の数よりも少なくしたことを特徴とする請求項 6 記載の画像データ生成方法。

【請求項 11】 前記第 1 の画像データが複数の色成分を有し、該色成分毎に、前記注目符号パターン群を複数用いて、該複数の注目符号パターン群と特定符号パターン群とを比較するとき、前記色成分の内、少なくとも 1 つの色成分については、前記注目符号パターン群の数を、他の色成分の注目符号パターン群の数よりも少なくしたことを特徴とする請求項 9 記載の画像データ生成方法。

【請求項 12】 前記少なくとも 1 つの色成分は黄色であることを特徴とする請求項 10 または 11 記載の画像データ生成方法。

【請求項 13】 第 1 の解像度の画像データから第 1 の解像度より高い第 2 の解像度の画像データを生成する機能をコンピュータに実現させるためのプログラムを記録したコンピュータ読み取り可能な記録媒体であって、前記第 1 の画像データの注目画素毎に、該画素の位置に対応した、ソートされた複数の閾値を用いて、画素値を階調数に変換する機能と、前記画素の位置および前記変換された階調数に対応した所定の符号パターンを参照することにより、前記第 2 の画像データを生成する機能をコンピュータに実現させるためのプログラムを記録したコンピュータ読み取り可能な記録媒体。

## 【発明の詳細な説明】

【0001】

【発明の属する技術分野】本発明は、画像情報を、出力装置用のデータとして中間調処理および解像度変換する画像データ生成方法および画像データ生成処理プログラムを記録した記録媒体に関し、アプリケーションプログラム、ディスプレイドライバ、プリンタドライバ等のデバイスドライバに適用される技術である。

【0002】

【従来の技術】コンピュータの普及により、テキスト、グラフィックス等のベクトルデータや、自然画等のイメージデータを含んだ複雑な文書が容易に作成できるようになった。このような文書を高品位でプリントできるように、プリンタの解像度も向上しているが、その結果、プリンタへ送信すべきデータ量が増大し、処理時間や転送時間の増大を招いている。

【0003】上記したような文書を所定の解像度および階調表現能力を有するプリンタを用いて出力する場合の

代表的な方法（以下、第1の方法）として、例えば特開平5-96792号公報に記載されているものがある。

【0004】この第1の方法では、プリンタドライバ内で、ベクトルデータを所定解像度のビットマップデータとしてラスター展開するとともに、自然画等のイメージデータに対して所定解像度への解像度変換を行った後、両者に対して所定階調数への階調数変換（中間調処理）を施し、階調変換後（中間調処理後）のデータをプリンタに対して送信するものである。従って、プリントスピードを向上させるためには、ラスター展開、解像度変換、および中間調処理の高速化が必要であり、またこれと同時に、プリンタへのデータ転送の高速化も必要である。

【0005】

【発明が解決しようとする課題】そこで、ラスター展開および中間調処理を高速化する方法が、「PPA Printer Software Driver Design」（HP Journal, June 1997）に記載されている。

【0006】この方法（以下、第2の方法）では、ラスター展開および中間調処理を高速化するため、ラスター展開およびイメージデータの解像度変換をプリンタの解像度よりも低い解像度で行い、中間調処理を施した後プリンタに送信し、プリンタのファームウェアを使って全体の解像度変換（イメージデータ単体の解像度変換と区別するため、これを以下“全体の”と呼ぶ）を行って、上記低解像度をプリンタの解像度に変換して出力する（なお、前掲した文献には、テキストをプリンタの1/2の解像度でラスター展開し、ファームウェアを使ってテキストの解像度を倍に変換することが記載されている）。

【0007】上記したように、ラスター展開、ビットマップの解像度変換および中間調処理を低解像度で行えば、処理すべきデータ量が削減される結果、高速化が図られる。しかし、中間調処理後のデータに解像度変換を施した場合、解像度変換を行ってから中間調処理を施す場合に比べて、明らかに画質が低下しやすい。

【0008】以上で説明した第1の方法、第2の方法を簡単に比較すると、以下のような傾向となり、概略「第1の方法は画質優先で、第2の方法は速度優先」と言える。

【0009】

ラスター展開 ; 第1の方法=遅い

第2の方法=速い

イメージの解像度変換 ; 第1の方法=遅い

第2の方法=速い

中間調処理 ; 第1の方法=遅い

第2の方法=速い

転送データ量 ; 第1の方法=多い

第2の方法=少ない

画質 ; 第1の方法=良い

第2の方法=悪い。

【0010】本発明は上記した事情を考慮してなされたもので、本発明の目的は、ラスター展開とイメージの解像度変換が従来の第1、第2の方法よりも速く、中間調処理、転送データ量、画質については従来の第1、第2の方法の中間となる、両者の長所を併せ持ち、中間調処理と全体の解像度変換を同時に行う画像データ生成方法および記録媒体を提供することにある。

【0011】すなわち、

ラスター展開 ; 第1の方法=遅い

本発明=速い

第2の方法=速い

イメージの解像度変換 ; 第1の方法=遅い

本発明=速い

第2の方法=速い

中間調処理 ; 第1の方法=遅い

本発明=第1と第2の中間

第2の方法=速い

転送データ量 ; 第1の方法=多い

本発明=第1と第2の中間

第2の方法=少ない

画質 ; 第1の方法=良い

本発明=第1と第2の中間

第2の方法=悪い。

【0012】

【課題を解決するための手段】前記目的を達成するために、請求項1記載の発明では、第1の解像度の画像データから第1の解像度より高い第2の解像度の画像データを生成する画像データ生成方法であって、前記第1の画像データの注目画素毎に、該画素の位置に対応した、ソートされた複数の閾値を用いて、画素値を階調数に変換し、前記画素の位置および前記変換された階調数に対応した所定の符号パターンを参照することにより、前記第2の画像データを生成することを特徴としている。

【0013】請求項2記載の発明では、前記階調数変換時に誤差拡散処理を適用することを特徴としている。

【0014】請求項3記載の発明では、同一階調値における前記符号パターンを、前記第1の画像データの画素位置に対応して配置したとき、該配置がドット分散型となることを特徴としている。

【0015】請求項4記載の発明では、同一階調値における前記符号パターンを、前記第1の画像データの画素位置に対応して配置したとき、該配置がブルーノイズ型となることを特徴としている。

【0016】請求項5記載の発明では、前記階調数変換をホスト側で行い、前記階調数を画像出力装置に対して送信し、前記階調数の送信がn進法で行われるとき、前

記階調数を $n$ のべき乗とすることを特徴としている。

【0017】請求項6記載の発明では、第1の解像度の画像データから第1の解像度より高い第2の解像度の画像データを生成する画像データ生成方法において、前記第1の画像データの注目画素毎に、該画素の位置に対応した、ソートされた複数の閾値を用いて、画素値を階調数に変換し、前記画素の位置および前記変換された階調数に対応した所定の第1の符号パターンを参照することにより、前記第2の画像データを生成する画像データ生成方法であって、前記注目画素に対応した符号パターンおよび所定個数の前記第1の画像データに対応した符号パターンから構成される所定形状の注目符号パターン群と、特定符号パターン群とを比較し一致したとき、前記注目符号パターン群の所定の第2の符号パターンを、第3の符号パターンに変更することを特徴としている。

【0018】請求項7記載の発明では、前記階調数変換をホスト側で行い、前記階調数を画像出力装置に対して送信し、該画像出力装置は前記階調数に対応した符号パターンを参照することにより前記第2の画像データを生成すると共に、前記第3の符号パターンへの変更を行うことを特徴としている。

【0019】請求項8記載の発明では、前記変更の対象となる前記第3の符号パターンは、前記所定の第1の符号パターンの何れとも異なることを特徴としている。

【0020】請求項9記載の発明では、前記注目符号パターン群を複数用いて、該複数の注目符号パターン群と特定符号パターン群とを比較することを特徴としている。

【0021】請求項10記載の発明では、前記第1の画像データが複数の色成分を有し、該色成分毎に、前記所定形状の注目符号パターン群と特定符号パターン群とを比較するとき、前記色成分の内、少なくとも1つの色成分については、前記特定符号パターン群の数を、他の色成分の特定符号パターン群の数よりも少なくしたことを特徴としている。

【0022】請求項11記載の発明では、前記第1の画像データが複数の色成分を有し、該色成分毎に、前記注目符号パターン群を複数用いて、該複数の注目符号パターン群と特定符号パターン群とを比較するとき、前記色成分の内、少なくとも1つの色成分については、前記注目符号パターン群の数を、他の色成分の注目符号パターン群の数よりも少なくしたことを特徴としている。

【0023】請求項12記載の発明では、前記少なくとも1つの色成分は黄色であることを特徴としている。

【0024】請求項13記載の発明では、第1の解像度の画像データから第1の解像度より高い第2の解像度の画像データを生成する機能をコンピュータに実現させるためのプログラムを記録したコンピュータ読み取り可能な記録媒体であって、前記第1の画像データの注目画素毎に、該画素の位置に対応した、ソートされた複数の閾

値を用いて、画素値を階調数に変換する機能と、前記画素の位置および前記変換された階調数に対応した所定の符号パターンを参照することにより、前記第2の画像データを生成する機能をコンピュータに実現させるためのプログラムを記録したコンピュータ読み取り可能な記録媒体であることを特徴としている。

【0025】

【発明の実施の形態】以下、本発明の一実施例を図面を用いて具体的に説明する。

（実施例1）図1は、本発明の実施例1の構成を示す。図において、1はラスター展開または解像度変換した後の画像データ、2は閾値が配列された閾値マトリックス、3は符号パターンが格納された符号パターン格納部、4はプリントデータであり、画像データ1と閾値マトリックスは処理装置5内にあり、符号パターン格納部3とプリントデータ4はプリンタ6内にある。また、本発明の実施例1では、例えば解像度600dpi、階調数2値のプリンタを出力装置として用いる。

【0026】以下、本発明を説明する前に前述した従来の第1、第2の方法について図を用いて説明する。図2は、階調数変換（中間調処理）で使用する $4 \times 4$ の閾値マトリックスを示す。マトリックスのデータの各画素は、例えば0から15までの値（各画素が4ビットの多値）をとる。

【0027】図3（a）は、前記した第1の方法によって、ベクトルデータを600dpiでラスター展開した後のデータ、あるいはイメージデータを600dpiに解像度変換した後のデータである。また、図4（a）は、同様に前記した第2の方法によって、300dpiでラスター展開または解像度変換した後のデータを示す。なお、この例では、図3（a）は、図4（a）を公知の最近傍法で2倍に解像度変換した場合と同じものになっている。

【0028】ラスター展開（あるいは解像度変換）自体の処理量は、概ね解像度の2乗に比例するので、第1の方法のラスター展開（あるいは解像度変換）の処理量は、第2の方法の処理量の4倍となる。

【0029】第1の方法においては、図3（a）のデータについて各画素値と、各画素位置に対応する図3（b）（図2と同一のマトリックス）の閾値マトリックスの閾値とを比較し、公知のディザ法によって2値化を行う。この例の場合、閾値以上の値を有する画素の値を1と出力するので、階調値として図3（c）を得るが、閾値に対して計16回の比較演算が必要となる。該データに対して、圧縮等を行わない場合には、図3（d）に示すデータがプリントすべきデータ（図3（c）と同一）となるため、プリンタに対しては、16画素 $\times$ 2bit=32bitのデータが転送されることになる。

【0030】同様に、第2の方法では、図4（a）のデータに対して、図2の閾値マトリックスを作用させる場

10

20

30

40

50

合は、 $4 \times 4$ のマトリックスの $1/4$ のマトリックスが順に使用され、この例では図4(b)のマトリックスによって2値化を行って、図4(c)の階調値を得る(図4(a)のデータの右に隣接する $2 \times 2$ のデータ(図は省略)に対しては、図2中で図4(b)のマトリックスの右に隣接する $2 \times 2$ のマトリックス(閾値3、15、11、7)が使用される)。

【0031】この第2の方法の場合、閾値に対して計4回の比較演算しか行われないので、第1の方法に比べて中間調処理に要する時間は少ない。また、データ圧縮等を行わない場合には、プリンタに対して4画素 $\times$ 2bit=8bitのデータが転送されることになり、転送データ量も少ない。

【0032】ただし、第2の方法においては、プリンタ側のファームウェアで、図4(c)のデータ全体の解像度を2倍にするため、プリントすべきデータは図4(d)となる。なお、この例では、解像度変換法として公知の最近傍法を用いる。

【0033】図5は、本実施例を説明する図であり、図5(a)は本実施例の画像データ1であり、前述した第2の方法と同様に、300dpiでラスター展開または解像度変換を行った後のデータ( $2 \times 2$ 画素)である。また、本実施例の閾値マトリックス2は、図5(b)のように閾値が配列されている。そして、画像データ1を、閾値マトリックス2によって0から4までの階調数に変換する。

【0034】図5(b)は、要素が4個の配列の $2 \times 2$ のマトリックスであり、配列を構成する4つの閾値は、図2のマトリックスを $2 \times 2$ のサブマトリックス4つに分解したときの各サブマトリックス内の閾値と同じであるが、閾値が値順にソートされている。

【0035】本実施例における階調数変換は、図5(a)の1画素のデータと、その画素位置に対応する4個の閾値とを比較して(つまり、図5(b)のソートされた閾値を順に比較して)、閾値が画素データを越えるまでの(比較回数-1)を階調値とする。すなわち、例えば、画像データ(15)と、その対応する閾値マトリックスの閾値(1、5、9、13)とを比較し、画像データ(7)と、それに対応する閾値マトリックスの閾値(3、7、11、15)とを比較する。

【0036】この結果、図5(a)のデータに対応する階調数として図5(c)が得られ、その比較回数は計11回となる。

【0037】また、本実施例においては、図5(c)に示した階調値をプリンタ6に対して送信するが、そのデータ量は、4画素 $\times$ 3bit(5階調)=12bitである。

【0038】そして、本実施例においては、プリンタ6側には、 $2 \times 2$ の符号パターンである5つの配列を要素とし、図5(a)の4つの画素の各位置に対応した $2 \times$

2の符号パターンマトリックス(図6)が符号パターン格納部3に設けられている。

【0039】図6の符号パターンマトリックスの各要素は、図2のマトリックスを $2 \times 2$ のサブマトリックス4つに分解し、各サブマトリックスに $2 \times 2$ のデータ(1画素16階調)を作用させたときに、とり得る $2 \times 2$ のパターンを、階調数順にソートしたものである。符号パターンは、ベクトル量子化におけるコードブックに相当するものである。

【0040】そして、図5(a)の値15の注目画素(\*印)に対しては、図6の中の太線内(\*印)の5つの符号パターンが対応し、ファームウェア中で、注目画素の階調値4に対応した二重線で囲まれた符号パターン(1、1、1、1)が選択される。他の画素も同様に、画素の位置と階調値に対応した符号パターンが選択され、最終的にプリントすべきデータ4として、図5(d)が得られる。

【0041】本実施例では、第2の方法で図4(d)の最終データを得るための全体の解像度変換に相当する処理を、符号パターンの参照によって行うため、“全体の解像度変換”にかかる処理は、第2の方法よりも高速ではない。しかし、図4(d)、3(d)、5(d)を比較すれば明らかなように、本実施例の方法による最終データ図5(d)は、第2の方法よりも高画質(例えば、原データの階調性がよりよく保存されている)で、第1の方法に匹敵するものである(本実施例では、簡単のため、図3と図4を最近傍法で解像度変換した関係にしたが、両者の関係がそうでない場合には、本実施例の最終データは第2の方法よりは劣ることになる)。

【0042】以上の例において、3つの方法の処理量の比較結果を簡単にまとめると、以下ようになる(中間調処理の処理量は、閾値との比較回数で近似している)。

【0043】

ラスター展開 ; 第1の方法=4

本発明=1

第2の方法=1

イメージの解像度変換 ; 第1の方法=4

本発明=1

第2の方法=1

中間調処理 ; 第1の方法=16

本発明=11

第2の方法=4

転送データ量 ; 第1の方法=32

本発明=12

第2の方法=8

全体の解像度変換 ; 第1の方法=不要

本発明=中速

第2の方法=高速

画質 ; 第1の方法=良い  
 本発明=中~良  
 第2の方法=悪い。

従って、本実施例は、第1の方法と第2の方法の両方の長所を併せ持った方法であるといえることができる。

【0044】図7は、本発明の実施例1の処理フローチャートである。x行×y列の画像データを、 $k^2$  ( $k$ の2乗)個のソートされた閾値の配列を要素とするm行×n列の閾値マトリックスで階調数変換して階調数を得て(ステップ102)、 $k \times k$ の符号パターン $k^2 + 1$ 個の配列を要素とするm行×n列の符号パターンマトリックスを参照して、 $k \times k$ 列のプリントデータを得る(ステップ103)。マトリックスの右端、下端に達したときは(ステップ106、110)、それぞれ画像データをn+1画素、m+1画素ずらした位置から同様に処理し、画像データの下端に至るまで(ステップ109)同じ処理を繰り返す。

【0045】図中、 $D(i, j)$ は、画像データ中の(行, 列) =  $(i, j)$ に位置するデータであり、 $M(p, q)$ は、閾値マトリックス中の(行, 列) =  $(p, q)$ に位置する閾値配列であり、 $t(i, j)$ は、 $D(i, j)$ に $M(p, q)$ を作用させたときに得られる階調数であり、 $P(p, q, t(i, j))$ は、 $D(i, j)$ に対して参照すべき、符号パターンマトリックス中の1つの符号パターンである。

【0046】また、ソートされた閾値の配列を要素とするm行×n列の閾値マトリックスは、 $k \times k$ 列の閾値マトリックスから図8の処理フローによって作成でき、m行×n列の符号パターンマトリックスは、 $k \times k$ 列の閾値マトリックスから、図9の処理フローによって作成できる。

【0047】なお、実施例1においては、原データの縦横比を維持するため、符号パターンの大きさを $k \times k$ としているが、維持の必要がない場合には、縦横比が1でない符号パターンを作成すればよい。

【0048】(実施例2) 実施例1では、 $k^2$ 個のソートされた閾値の配列を用いて $k^2 + 1$ 階調への変換を行った。通常、プリンタへのデータ転送は2進法で行われるため、ここで階調数が2のべき乗になるように構成すると、(転送データ/階調数)が最小となり転送効率がよい。

【0049】図10は、実施例2を構成する閾値マトリックスであり、3個のソートされた閾値の配列を用いて4階調(0から3)への変換を行えば、4画素分の階調データは8bit (= 4画素×2bit (4階調))で済むため、(転送データ/階調数)を実施例1よりも小さくすることができる。なお、図10の各サブマトリックスは、図5(b)の各サブマトリックスの閾値配列の中央2つの値を、該2つの値の平均値1つで置き換えたものである。

【0050】(実施例3) 上記した実施例は、ディザ法による中間調処理を第2の方法よりも高速に行うものであるが、公知の誤差拡散法にも適用可能である。図11は、本発明を誤差拡散法に適用した場合の実施例3を説明する図である。図11(a)は、実施例3で使用する周知の誤差マトリックスであり、\*印の注目画素(図5(a)の画素値15)の階調数変換で生じた誤差(15-13=2)を、図11(b)のように注目画素に隣接する右および下の画素に半分ずつ加算し、図11(c)に示すデータの注目画素(\*印)に対して、さらに階調数変換を行う。本実施例でも、図6と同様な符号パターンマトリックスが使用できる。図12は、実施例3の処理フローを示し、図7の処理にさらに誤差を配分するステップ204が追加されている。

【0051】(実施例4) 実施例1は、図2のマトリックスを基に符号パターンマトリックスを生成したが、図13に示すような典型的なドット分散型の閾値マトリックスを基に符号パターンマトリックスを生成すると、図14のように、同一階調値における符号パターンの配列自体もドット分散型となる。すなわち、図14において、例えば階調値1に対応した4つの符号パターンを、原画像データの画素位置に対応させて配置すると、 $4 \times 4$ のマトリックスができるが、これはドット分散型である。なお、図2のマトリックスもドット分散的な性質を有する。

【0052】図14から明らかなように、符号パターンマトリックスを構成する符号パターンの配列は全て同じとなるから、図14の例の場合、プリンタ側で5種類の符号パターンのみを保持することにより、符号パターンマトリックスの容量を小さくできると同時に、マトリックスの参照も高速となる。

【0053】(実施例5) また、本発明では、図示しない公知のブルーノイズ型の閾値マトリックス(閾値の配列がランダム)を基に符号パターンマトリックスを生成することも可能である。この場合には、同一階調値における符号パターンの配列自体もブルーノイズ型となる。そして、本実施例では、ディザ法と同様の処理速度で、誤差拡散に匹敵し得る画質を得ることができる。

【0054】なお、ブルーノイズ型の閾値マトリックスに関しては、「Digital Halftoning Using a Blue Noise Mask」, T. Mitsa and K. Parker, 1991 International Conference on Acoustics, Speech, and Signal Processing, February and May 1991 (ブルーノイズマスク法の原理など) および「Modified Approach to the Construction of a Blue Noise Mask」by M. Y. ao and K. Parker, Journal of



Electronic Imaging, Vol. 3, No. 1, 92-97, January 1994 (ブルーノイズマスクの作成法等)を参照されたい。

【0055】(実施例6)上記した実施例では、プリンタ6側が符号パターンマトリックス3を保持していたが、処理装置5(ホスト)側から符号パターンマトリックス3をダウンロードするように構成してもよい。また、階調数変換に使用する閾値マトリックスを切り替える場合には、符号パターンの変更を伴う。これらの場合ホストが、階調データの送信に先立ち、符号パターンマトリックスをプリンタに送信すればよい。

【0056】図15は、実施例6の処理フローチャートである。プリントを開始する前に、ホスト5は符号パターンが送信済みでなく(ステップ301)、符号パターンに変更があるとき(ステップ302)、符号パターンをプリンタ6に送信し(ステップ303)、その後、ホストがプリンタに階調値を送信し(ステップ304)、プリンタ6は送信された階調値に対応した符号パターンを参照してプリントデータを出力する(ステップ305)。

【0057】このように、本実施例では、階調数をプリンタなどの画像出力装置に対して送信するので、転送データ量が削減でき、また、符号パターンをダウンロード可能にしたので、符号パターンの変更に対応することができる。

【0058】(実施例7)ところで、前述した第2の方法は、例えば特開平2-112966号公報に示されるようなビットパターンのテンプレートマッチングを行うことによって、ジャギーのスムージングを行うことが可能である。しかし、ビットパターン同士でテンプレートマッチングを行う場合には、処理量が多くなり、速度低下を招きやすいという問題がある。

【0059】そこで、本実施例では、後述する符号パターンのテンプレートマッチングを行い、簡易な処理を行うことによって、高速にジャギーのスムージングを行う方法を提案する。

【0060】以下、本実施例について説明するが、本実施例においても、実施例1と同様に出力装置として解像度600dpi、階調数2値のプリンタを用いる。また、想定する閾値マトリックスは図2と同様であり、図14と同じ符号パターンマトリックスを使用する。ただし、本実施例では、図16に示すように、異なる符号パターンにパターン番号(0、1、2、3a、3b、4)を付与したものを用いる。また、原画像データとしては、黒色のみのモノクロ画像を想定する。

【0061】さて、図17、18は、「文」という黒文字のベクトルフォントを300dpiでラスター展開し、実施例1の処理手順によって得られたビットパターン(符号パターン)を示す(なお、本来は1枚の図であるが、これを2つに分割し、図17は「文」の左半分で

あり、図18は「文」の右半分である)。グレーの部分には符号パターン中の値1が、白い部分には符号パターン中の値0が対応していて、この例では濃い黒色の文字の処理結果であることから、階調値としては0または4のいずれかとなっている。

【0062】図17の左上(a)には、300dpi、600dpiの各画素のサイズが示されていて、(b)には、600dpiの画素サイズに対応した2×2の符号パターンの並び、およびパターン番号の並びが示されている。また、(b)中には、本実施例における、符号パターンのサイズおよび符号パターン群のサイズが示されている。

【0063】この例においては、注目符号パターンを含む4つの符号パターンから成る正方形の注目符号パターン群を、6種類の正方形の特定符号パターン群(図19)と照合し、一致した場合には、注目符号パターン群をそれぞれ図20に示す符号パターン群に変更する。図20において黒色で示した部分は、符号パターン中の変更された位置を示すが、図16から明らかなように、図20の変更後の符号パターン(黒色を含む符号パターン)の内の3つは、図16には存在しない。つまり、図20(e)の符号パターンは、図16のパターン番号1として存在するが、図20(c)、(d)、(f)の符号パターンは、図16のパターン中にはない。

【0064】従って、本実施例では、図21に示す3つの符号パターン(パターン番号1b、1c、1d)を追加して使用する。また、図16から明らかなように、符号パターンには重複が生じるので、予め重複を排除したパターン番号を図16、図21に示すように定義する。なお、符号パターン数の増加が好ましくない場合には、符号パターンを追加しないという選択も可能である。

【0065】ここで、図19(a)の注目符号パターン群を例にとると、図19と図20の関係は以下の通りである。図19(a)の特定符号パターン群は、第1象限から左回りに0、4、0、4というパターン番号の順列を形成する。そこで、注目符号パターン群が、第1象限から左回りに0、4、0、4というパターン番号列を形成した場合には、これを図20(a)に示す1d、4、1b、4というパターン番号列(符号パターン群)に変更する。

【0066】また、この例においては、1つの注目符号パターン(図17(c)の\*印)に対する注目符号パターン群(比較のためのウインドウ)のとり方として、図17(c)に示すA、B、C、Dの4種類(太線A、二重線B、点線C、三重線D)を採用し、注目符号パターン自体のとり方としては、図17(d)の\*印に示す通り、縦方向・横方向ともに1つ置きとしている(1つ置きで十分だからである)。図17、18全体に配置された格子は、図17(d)の\*印に対する、注目符号パターン群Aのとりかたと一致している。

【0067】本実施例においては、1つの注目符号パターンに対し、注目符号パターン群（比較のためのウィンドウ）Aをとり、特定符号パターン群（図19）と一致した場合には注目符号パターン群を図20に変更して、さらに注目符号パターン群Bと比較し、といった処理を繰り返す。

【0068】なお、本実施例においては、注目符号パターン群Aにおける符号パターンの変更が、注目符号パターン群Bにおける比較・変更に影響を与えないため、1つの注目符号パターンに対してA、B、C、Dととって比較・変更した後、次の注目符号パターンに移るという処理方法でも、全ての注目符号パターンに対してAをとって比較し、全ての注目符号パターンに対してBをとって比較し、という処理方法でも結果は同じである。

【0069】そこで、図22は、全ての注目符号パターンについて、注目符号パターン群（比較のためのウィンドウ）Aをとった場合の処理結果を示したものであり、黒い部分に変更された符号パターンを示す。これによって、ジャギーのスムージングがなされ、比較のためのウィンドウが1つだけの場合でもある程度の効果があることが分かる。

【0070】同様に、図23は、全ての注目符号パターンについて、注目符号パターン群（比較のためのウィンドウ）Bをとった場合の処理結果を示したものであり、図17、18全体に配置された格子は、図17（d）の＊印に対する、注目符号パターン群Bのとりかたと一致している。図中の11から25部分が新たに変更された符号パターンを示す。

【0071】同様に、注目符号パターン群（比較のためのウィンドウ）Cをとった処理結果が図24、注目符号パターン群（比較のためのウィンドウ）Dをとった処理結果が図25図であり、それぞれ31から48の部分、51から62の部分の部分が新たに変更された符号パターンを示す。比較のためのウィンドウを複数使用する分、スムージングの効果も大きくなっているのが明らかである。

【0072】このように本実施例によれば、簡単かつわずかに6つの特定パターンの比較・検出という簡易な方法によって、かなりのスムージング効果が発揮されることがわかる（もちろん、スムージングによって弊害が生じる部分も存在するが、全体の効果からみればわずかなものである）。そして、簡易な方法であるので、高速にスムージングを行うことが可能になる。

【0073】なお、原画像データの端部においては、注目符号パターン群（比較のためのウィンドウ）を4つ全てとることができない場合があるが、その場合はとり得る注目符号パターン群のみ比較すればよい。

【0074】図26は、以上で説明した実施例7の処理フローチャートである。まず、実施例1で説明した方法によって、注目画素に対応する階調数を得る（ステップ401）。上記した階調数変換をホスト側で行い、階調

数をホストからプリンタに送信する（ステップ402）。

【0075】以下の処理はプリンタによって行われる。すなわち、階調数に対応した符号パターンマトリックス（図16）を参照して、注目画素のパターン番号を得る（ステップ403）。上記した処理を全ての注目画素について行う（ステップ404）。

【0076】注目符号パターンについて、A～Dの全ての注目符号パターン群と全ての特定符号パターン群（図19）を比較し（ステップ405）、一致したとき（ステップ406）、注目符号パターン群中の所定符号パターン（パターン番号）を変更する（ステップ407）。

【0077】例えば、注目画素のパターン番号が0（白）で、注目符号パターンが図19の（f）に一致したとき、注目画素のパターン番号0を、図20（f）に変更し、つまりパターン番号1b（図21）に変更する。また、例えば、注目画素のパターン番号が4（黒）で、注目符号パターンが図19の（d）に一致したとき、注目符号パターン中のパターン番号0を、パターン番号1d（図21）に変更する。上記した処理を全ての注目符号パターンについて行う（ステップ408）。

【0078】注目画素のパターン番号（図16）と変更されたパターン番号（図21）から、注目画素に対応する600dpiの最終画像データを得る（ステップ410）。

【0079】（実施例8）上記した実施例では、黒色のみのモノクロ画像を想定したが、カラー画像の場合には公知のように、以上の処理を各画像成分毎に行えばよい。すなわち、300dpiでラスター展開または解像度変換を行った後のデータを、公知の方法で各色成分、例えばC（シアン）、M（マゼンタ）、Y（黄色）、K（黒）毎のプレーンに分離し、各プレーン毎に図26の処理を行えばよい。

【0080】ここで、例えば上記4プレーンの全てに関して、6つの特定符号パターン群全て（図19）を用いた比較を行っても良いが、スムージング効果よりも高速性を求める場合、視認性の低い黄色については、特定符号パターン群の数を減らしてもよい。すなわち、特定符号パターン群のサブセットを使用する。例えば、黄色に関しては、図19（a）、（b）に示した2つの特定符号パターン群との比較のみに止める。この場合の処理フローは、黄色のみに関して、図26における比較ステップ405を「注目符号パターンについて全ての注目符号パターン群と、特定符号パターン群のサブセットを比較（パターン番号列同士を比較）」と置き換えればよい。

【0081】あるいは、黄色に関しては、上記4つの注目符号パターン群A、B、C、D（図17（c））を用いた比較を行わずに、Aのみを用いた比較に止めても良い。前述したように、比較のためのウィンドウが1つだけの場合でもスムージング効果があるからである。この

場合の処理フローは、黄色のみに関して、図26における比較ステップ405を「注目符号パターンについて、注目符号パターン群Aと、全ての特定符号パターン群を比較（パターン番号列同士を比較）」と置き換えればよい。

【0082】上記した例では、視認性の低い黄色について注目符号パターン群の数を少なくしたので、ユーザによって観察されるスムージング効果の低下を抑えながら処理速度を向上させることができる。そして、処理速度を最優先させるならば、黄色については特定符号パターン群との比較を行わないという選択も可能である。

【0083】なお、上記した実施例では文字への適用について示したが、本発明はこれに限らず、グラフィックス画像や自然画等にも適用可能である。

【0084】（実施例9）本発明は上記した実施例に限定されず、ソフトウェアによっても実現することができる。本発明をソフトウェアによって実現する場合には、図27に示すように、CPU、メモリ、表示装置、ハードディスク、キーボード、CD-ROMドライブ、スキャナなどからなるコンピュータシステムを用意し、CD-ROMなどのコンピュータ読み取り可能な記録媒体には、本発明の画像データ生成機能を実現するプログラムなどが記録されている。そして、本発明の画像データ生成処理を実行させるときは、上記した記録媒体に記録されたプログラムを読み出し、これをプリンタドライバとしてOSの一部に組み込むことによって本発明の処理機能が実現される。また、処理対象となる原画像データは例えばハードディスクに格納されていて、これに上記した処理が施され、原画像データよりも解像度の高い画像データがプリンタなどに出力される。

【0085】

【発明の効果】以上、説明したように、請求項1、13記載の発明によれば、原画像データの注目画素に対して、ソートされた閾値を用いて階調数変換を行い、符号パターンを参照するので、原画像データよりも解像度の高い画像データを高速に生成することができ、従来の第2の方法よりも高画質な画像を得ることができる。

【0086】請求項2記載の発明によれば、階調数変換時に誤差拡散処理を適用するので、高画質な画像データを生成することができる。

【0087】請求項3記載の発明によれば、同一階調値における符号パターンのドット配置がドット分散型の配置をとるようにしたので、符号パターンの数を少なくすることができる。

【0088】請求項4記載の発明によれば、同一階調値における符号パターンのドット配置がブルーノイズ型の配置をとるようにしたので、高速かつ誤差拡散処理に匹敵する高画質な画像データを生成することができる。

【0089】請求項5記載の発明によれば、階調数の送信がn進法でなされる場合に、階調数をnのべき乗とし

たので、階調数当たりの転送データ量を最小にすることができる。

【0090】請求項6記載の発明によれば、テンプレートマッチングによって、注目符号パターン群の所定の第2の符号パターンを、第3の符号パターンに変更するので、原画像データよりも解像度が高かつスムージングのかかった画像データを高速に生成することができる。

【0091】請求項7記載の発明によれば、階調数変換をホストコンピュータ側で行い、階調数を画像出力装置に対して送信し、画像出力装置が符号パターンを参照して原画像データよりも解像度の高い画像データを生成すると共に、第3の符号パターンへの変更を行うので、ホストからプリンタへの転送データ量が削減でき、かつスムージングのかかった最終画像データを高速に生成することができる。

【0092】請求項8記載の発明によれば、変更の対象となる第3の符号パターンが、画素に対応した符号パターンのいずれとも異なるので、符号パターンが拡張される結果、高いスムージング効果を得ることができる。

【0093】請求項9記載の発明によれば、注目符号パターン群を複数用いるので、高いスムージング効果を得ることができる。

【0094】請求項10記載の発明によれば、原画像データの色成分の少なくとも1つについて、特定符号パターン群の数を少なくしたので、スムージング効果への影響を抑えながら処理速度を高めることができる。

【0095】請求項11記載の発明によれば、原画像データの色成分の少なくとも1つについて、注目符号パターン群の数を少なくしたので、スムージング効果への影響を抑えながら、処理速度を高めることができる。

【0096】請求項12記載の発明によれば、1つの色成分を視認性の低い黄色としたので、ユーザによって観察されるスムージング効果の低下を抑えながら、処理速度を向上させることができる。

【図面の簡単な説明】

【図1】本発明の実施例1の構成を示す。

【図2】階調数変換（中間調処理）で使用する4×4の閾値マトリックスを示す。

【図3】（a）～（d）は、従来の第1の方法を説明する図である。

【図4】（a）～（d）は、従来の第2の方法を説明する図である。

【図5】（a）～（d）は、本実施例を説明する図である。

【図6】符号パターンマトリックスを示す。

【図7】本発明の実施例1の処理フローチャートである。

【図8】本発明の実施例1の閾値マトリックス作成の処理フローチャートである。

【図9】本発明の実施例1の符号パターンマトリックス

作成の処理フローチャートである。

【図10】本発明の実施例2を構成する閾値マトリックスである。

【図11】本発明を誤差拡散法に適用した場合の実施例3を説明する図である。

【図12】本発明の実施例3の処理フローチャートである。

【図13】ドット分散型の閾値マトリックスを示す。

【図14】本発明の実施例4の符号パターンマトリックスを示す。

【図15】本発明の実施例6の処理フローチャートである。

【図16】本発明の実施例7で使用する符号パターンマトリックスを示す。

【図17】「文」という黒文字のベクトルフォントをラスタ展開し、実施例1の処理手順によって得られた符号パターンの左半分を示す。

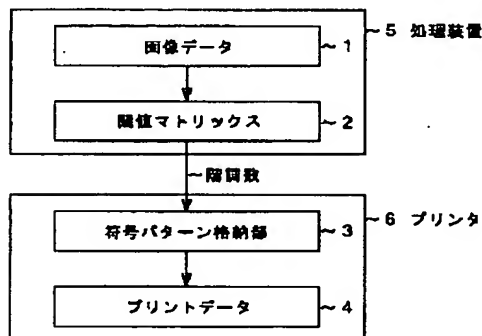
【図18】「文」の右半分を示す。

【図19】(a)～(f)は、特定符号パターン群を示す。

【図20】(a)～(f)は、変更後の符号パターン群を示す。

【図21】追加された符号パターンとパターン番号を示\*

【図1】



【図10】

1, 7, 13	3, 9, 15
4, 10, 16	2, 8, 14

\*す。

【図22】図17、18の全ての注目符号パターンについて、比較のためのウィンドウAをとった場合の処理結果を示す。

【図23】図17、18の全ての注目符号パターンについて、比較のためのウィンドウBをとった場合の処理結果を示す。

【図24】図17、18の全ての注目符号パターンについて、比較のためのウィンドウCをとった場合の処理結果を示す。

【図25】図17、18の全ての注目符号パターンについて、比較のためのウィンドウDをとった場合の処理結果を示す。

【図26】本発明の実施例7の処理フローチャートである。

【図27】本発明の実施例9の構成を示す。

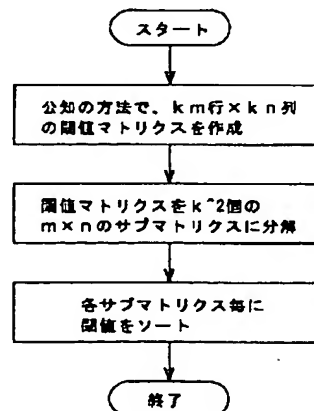
【符号の説明】

- 1 画像データ
- 2 閾値マトリックス
- 3 符号パターン格納部
- 4 プリントデータ
- 5 処理装置
- 6 プリンタ

【図2】

1	9	3	15
13	5	11	7
4	16	2	10
12	8	14	6

【図8】



【図3】

データ				関連マトリクス			
15	15	7	7	1	9	3	16
15	15	7	7	13	6	11	7
1	1	4	4	4	16	2	10
1	1	4	4	12	8	14	6

(a)

(b)

【図4】

15	7	1	9
1	4	13	5

(a)

(b)

階調値				最終データ			
1	1	1	0	1	1	1	0
1	1	0	1	1	1	0	1
0	0	1	0	0	0	1	0
0	0	0	0	0	0	0	0

(c)

(d)

1	0	1	1	0	0
0	0	0	0	0	0

(c)

(d)

【図5】

データ		関連マトリクス	
15	7	1, 6, 9, 13	3, 7, 11, 16
*			
1	4	4, 8, 12, 16	2, 6, 10, 14

(a)

(b)

階調値		最終データ	
4	2	1	1
0	1	1	0
		0	0
		0	0

(c)

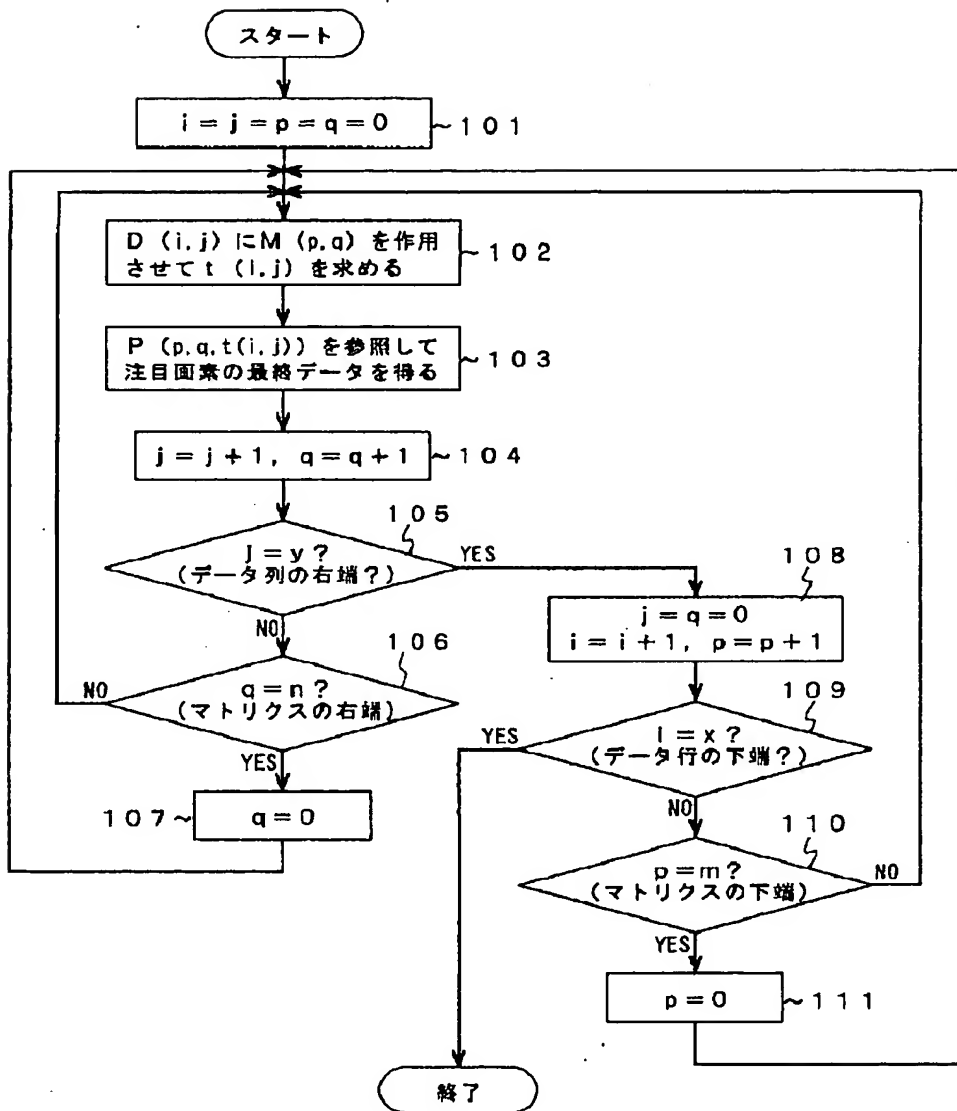
(d)

【図6】

符号パターンマトリクス

階調値	符号パターン	階調値	符号パターン
0	0 0 0 0	0	0 0 0 0
1	1 0 0 0	1	1 0 0 0
2	1 0 0 1	2	1 0 0 1
*			
3	1 1 0 1	3	1 0 1 1
4	1 1 1 1	4	1 1 1 1

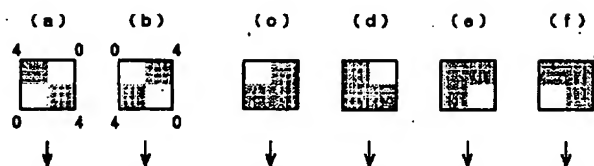
【図7】



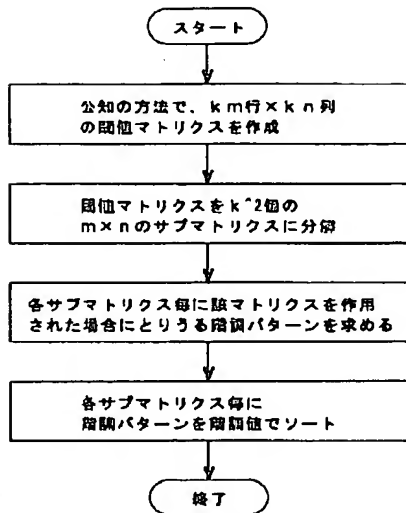
【図13】

1	9	3	11
13	5	16	7
4	12	2	10
16	8	14	6

【図19】



【図9】



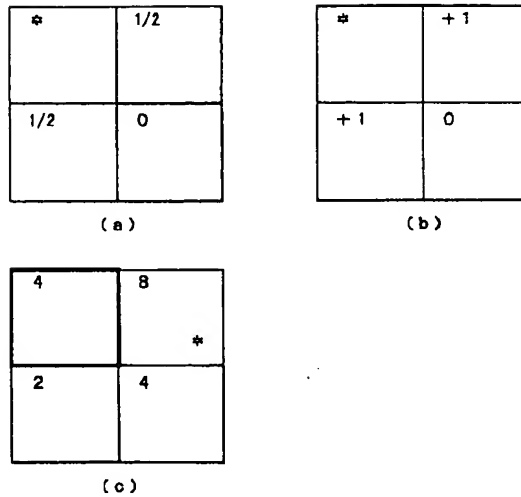
【図14】

符号パターンマトリクス

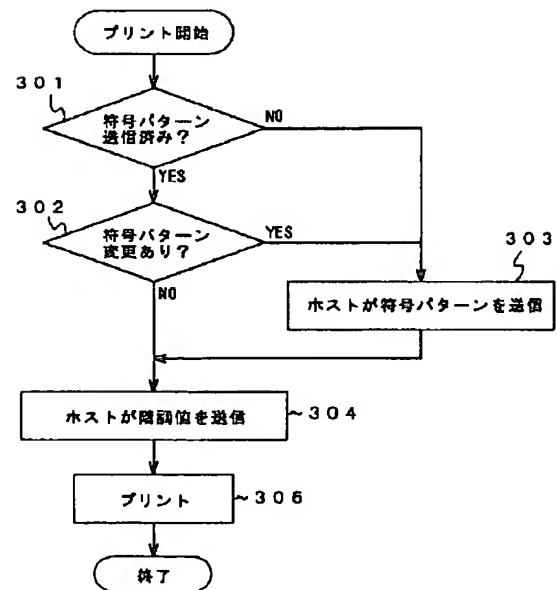
階関値	符号 パターン	階関値	符号 パターン								
0 →	<table><tr><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td></tr></table>	0	0	0	0	0 →	<table><tr><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td></tr></table>	0	0	0	0
0	0										
0	0										
0	0										
0	0										
1 →	<table><tr><td>1</td><td>0</td></tr><tr><td>0</td><td>0</td></tr></table>	1	0	0	0	1 →	<table><tr><td>1</td><td>0</td></tr><tr><td>0</td><td>0</td></tr></table>	1	0	0	0
1	0										
0	0										
1	0										
0	0										
2 →	<table><tr><td>1</td><td>0</td></tr><tr><td>0</td><td>1</td></tr></table>	1	0	0	1	2 →	<table><tr><td>1</td><td>0</td></tr><tr><td>0</td><td>1</td></tr></table>	1	0	0	1
1	0										
0	1										
1	0										
0	1										
3 →	<table><tr><td>1</td><td>1</td></tr><tr><td>0</td><td>1</td></tr></table>	1	1	0	1	3 →	<table><tr><td>1</td><td>1</td></tr><tr><td>0</td><td>1</td></tr></table>	1	1	0	1
1	1										
0	1										
1	1										
0	1										
4 →	<table><tr><td>1</td><td>1</td></tr><tr><td>1</td><td>1</td></tr></table>	1	1	1	1	4 →	<table><tr><td>1</td><td>1</td></tr><tr><td>1</td><td>1</td></tr></table>	1	1	1	1
1	1										
1	1										
1	1										
1	1										

階関値	符号 パターン	階関値	符号 パターン								
0 →	<table><tr><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td></tr></table>	0	0	0	0	0 →	<table><tr><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td></tr></table>	0	0	0	0
0	0										
0	0										
0	0										
0	0										
1 →	<table><tr><td>1</td><td>0</td></tr><tr><td>0</td><td>0</td></tr></table>	1	0	0	0	1 →	<table><tr><td>1</td><td>0</td></tr><tr><td>0</td><td>0</td></tr></table>	1	0	0	0
1	0										
0	0										
1	0										
0	0										
2 →	<table><tr><td>1</td><td>0</td></tr><tr><td>0</td><td>1</td></tr></table>	1	0	0	1	2 →	<table><tr><td>1</td><td>0</td></tr><tr><td>0</td><td>1</td></tr></table>	1	0	0	1
1	0										
0	1										
1	0										
0	1										
3 →	<table><tr><td>1</td><td>1</td></tr><tr><td>0</td><td>1</td></tr></table>	1	1	0	1	3 →	<table><tr><td>1</td><td>1</td></tr><tr><td>0</td><td>1</td></tr></table>	1	1	0	1
1	1										
0	1										
1	1										
0	1										
4 →	<table><tr><td>1</td><td>1</td></tr><tr><td>1</td><td>1</td></tr></table>	1	1	1	1	4 →	<table><tr><td>1</td><td>1</td></tr><tr><td>1</td><td>1</td></tr></table>	1	1	1	1
1	1										
1	1										
1	1										
1	1										

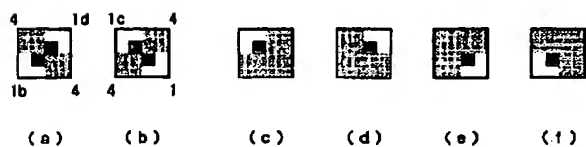
【図11】



【図15】



【図20】



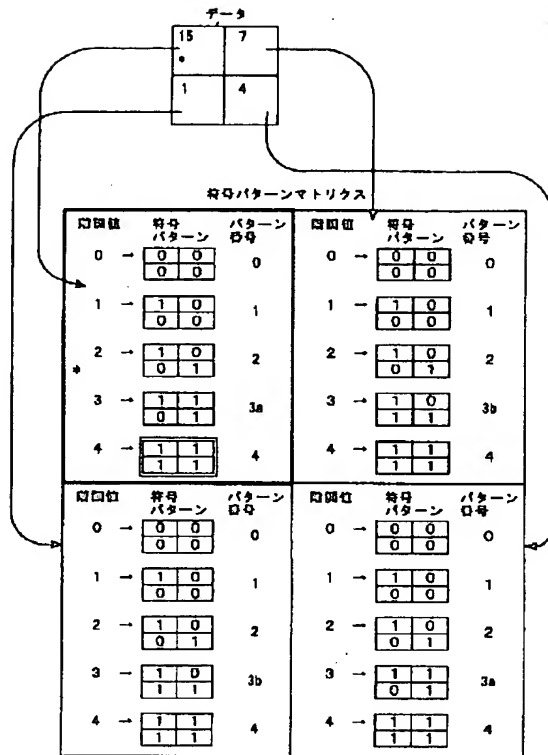
```

graph TD
    200[0] --> 201[~ 201]
    201 --> 202[を作用  
める]
    202 --> 203[照して  
を帰る]
    203 --> 204[配分]
    204 --> 205[+ 1]
    205 --> 206[206]
    206 --> 207{?}
    207 -- YES --> 209[209]
    207 -- NO --> 208{端)}
    209 --> 210[j = a = 0  
l = l + 1, p = p + 1]
    210 --> 211{l = x ?  
(データ行の下端?)}
    211 -- YES --> 208
    211 -- NO --> 212{p = m ?  
(マトリクスの下端)}
    212 -- YES --> 213[p = 0]
    212 -- NO --> 208
    213 --> 214[終了]
    208 --> 214
  
```

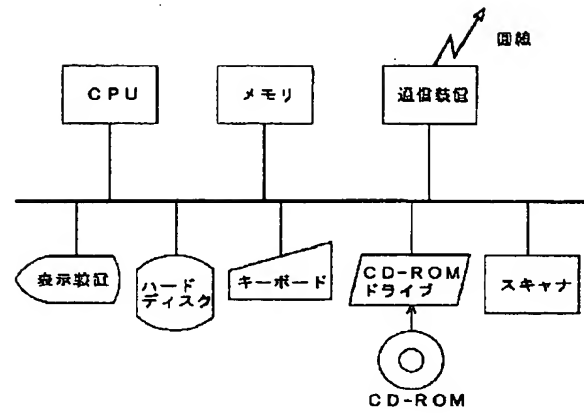
追加符号 パターン	パターン 番号				
<table border="1"> <tr><td>0</td><td>1</td></tr> <tr><td>0</td><td>0</td></tr> </table>	0	1	0	0	1 b
0	1				
0	0				
<table border="1"> <tr><td>0</td><td>0</td></tr> <tr><td>0</td><td>1</td></tr> </table>	0	0	0	1	1 c
0	0				
0	1				
<table border="1"> <tr><td>0</td><td>0</td></tr> <tr><td>1</td><td>0</td></tr> </table>	0	0	1	0	1 d
0	0				
1	0				



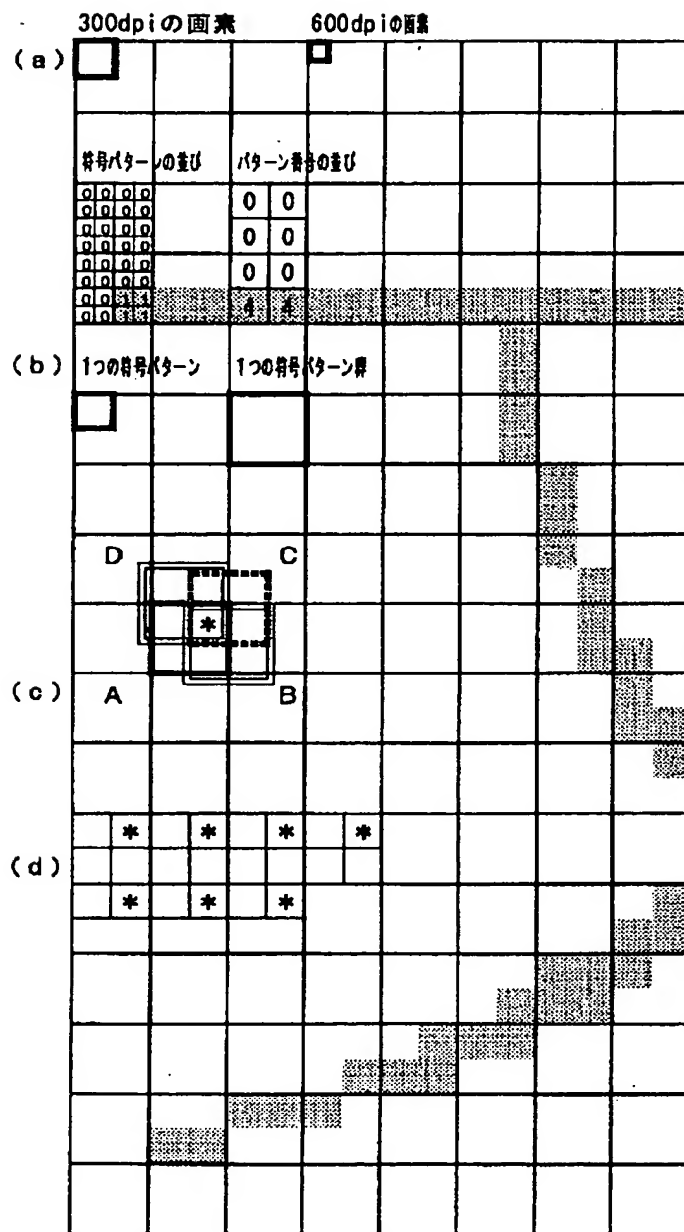
【図16】



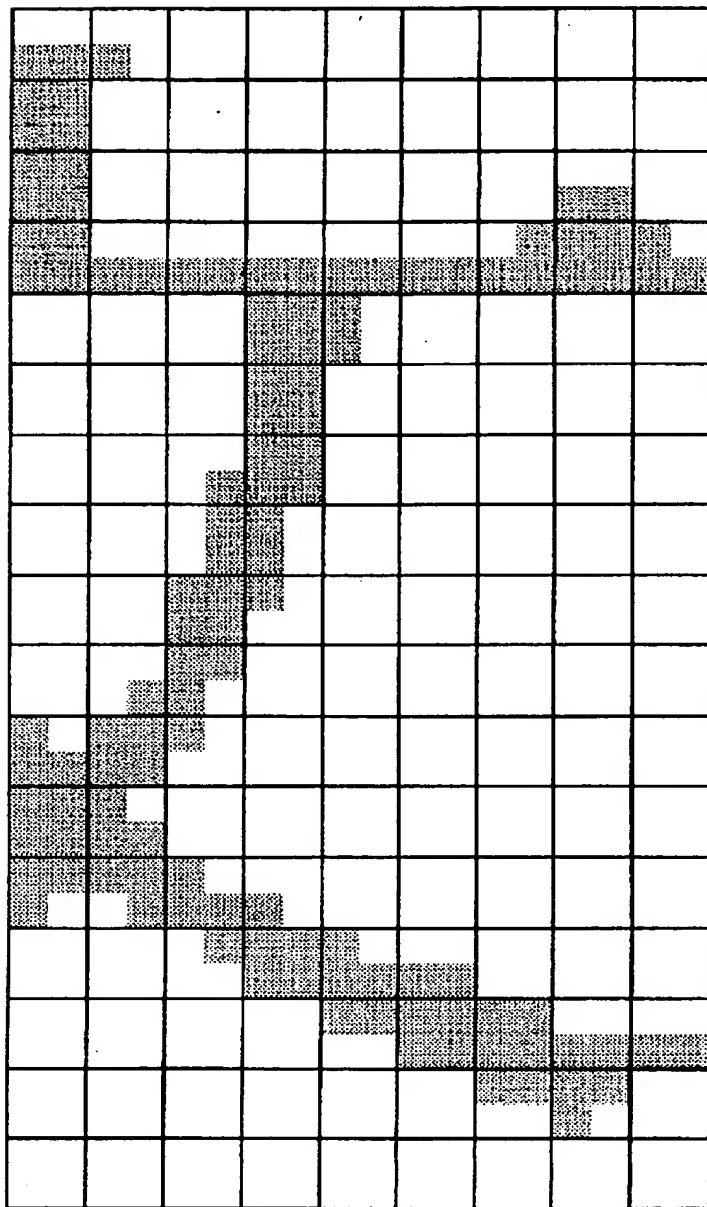
【図27】



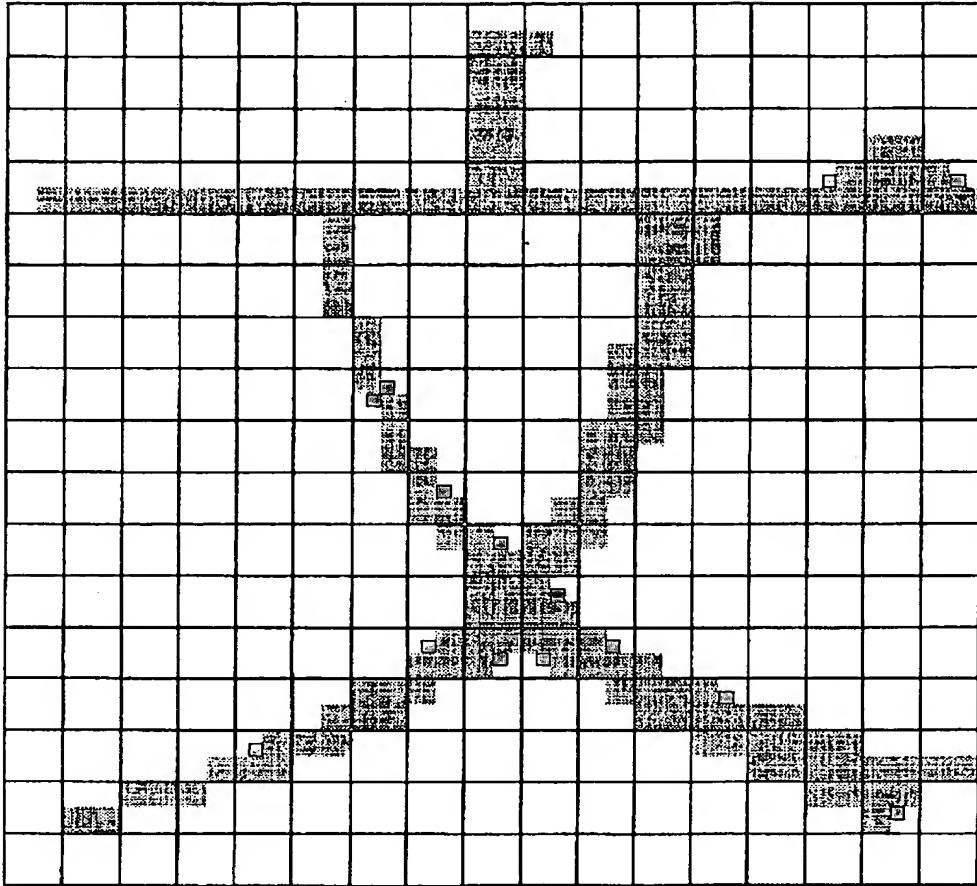
【图 17】



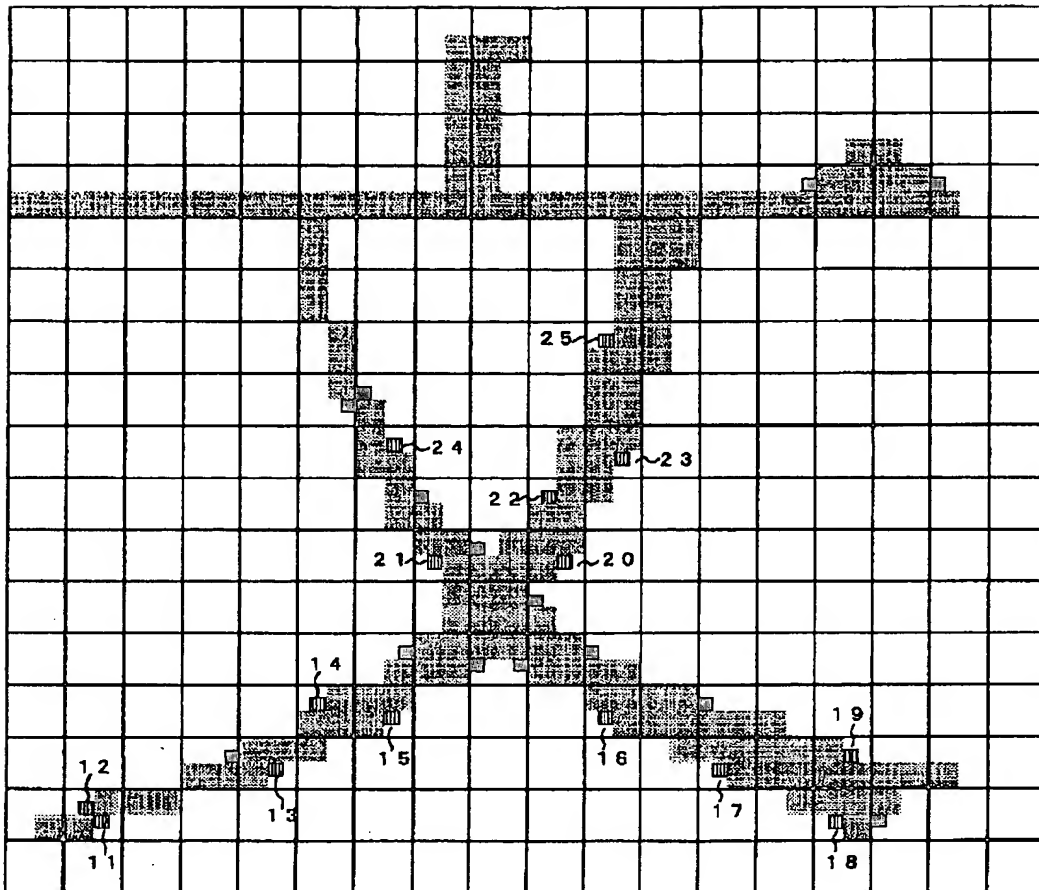
【図18】



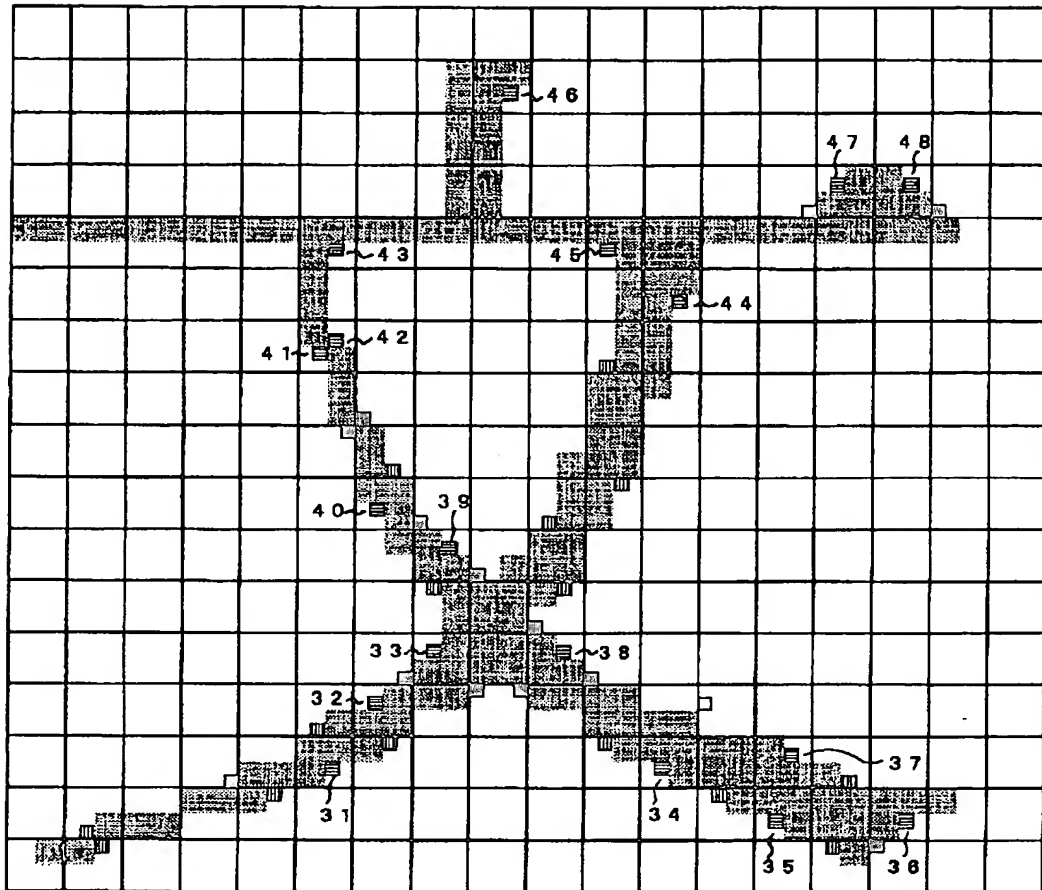
【図22】



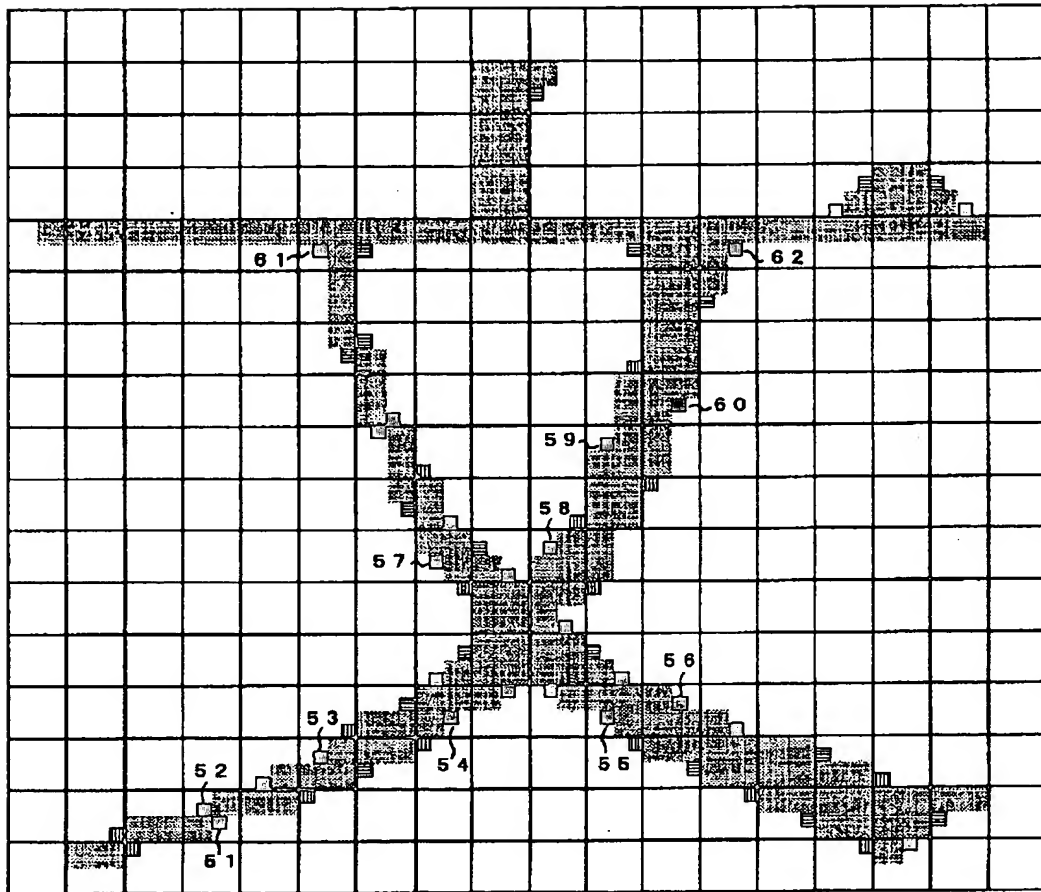
【図23】



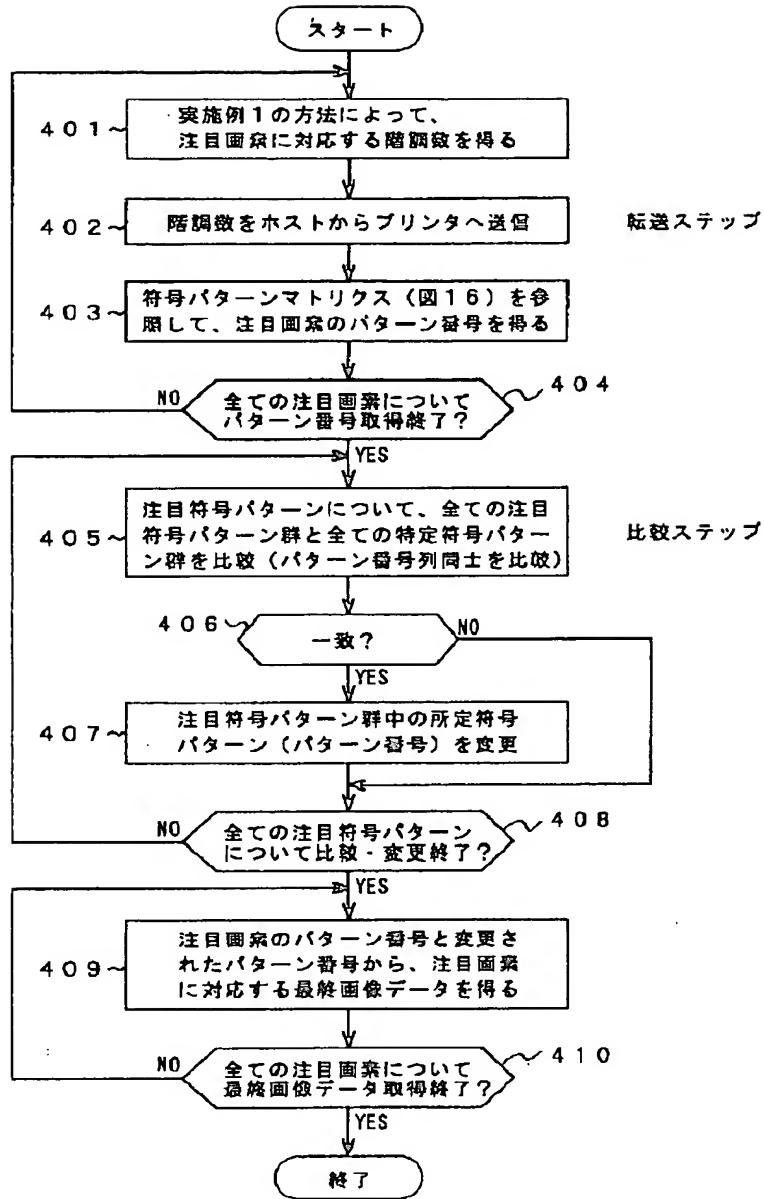
【図24】



【図25】



【図26】





フロントページの続き

F ターム(参考) 5B057 BA29 BA30 CA01 CA02 CA08  
CA12 CA16 CB01 CB02 CB08  
CB12 CB16 CB18 CC02 CD05  
CE13 CE16 CH07 CH18 DA17  
5C076 AA21 AA27 BA06 BA07  
5C077 LL18 MP01 MP08 NN15 PP20  
PP55 PP68 PQ08 PQ20 RR04  
RR08 RR16